OrangeFS For The Clouds

Walt Ligon

Clemson University

INTRODUCTION

Why Parallel File Systems?

- HPC and Big Data applications increasingly rely on I/O subsystems
  - Large input datasets, checkpointing, visualization
- Programmers need interfaces that match their problem
  - Multidimensional arrays, typed data, portable formats
- Two issues to be resolved by I/O system
  - Performance requirements (concurrent access to HW)
  - Gap between app. abstractions and HW abstractions
- Software is required to address both of these problems

What is OrangeFS?

- OrangeFS is a next generation Parallel File System
  - Based on PVFS
  - Distributes file data across multiple file servers leveraging any block level file system.
  - Distributed Meta Data across all servers using Berkley DB
  - Supports simultaneous access by multiple clients, including Windows
  - Works w/ standard kernel releases and does not require custom kernel patches
  - Easy to install and maintain

PVFS to OrangeFS

1994-2004

PVFS 1.0

Design and Development at
CU Dr. Ligon + ANL (CU Graduates)

2004-2010

PVFS 2.0

Primary Maint & Development
ANL (CU Graduates) + Community

2007-2010

OrangeFS PVFS Branch
initial development at CU + OB

SC10 (fall 2010)

PVFS 2.8

Announced to community and is now
Mainline of PVFS development
Commercial Grade Services available

2012

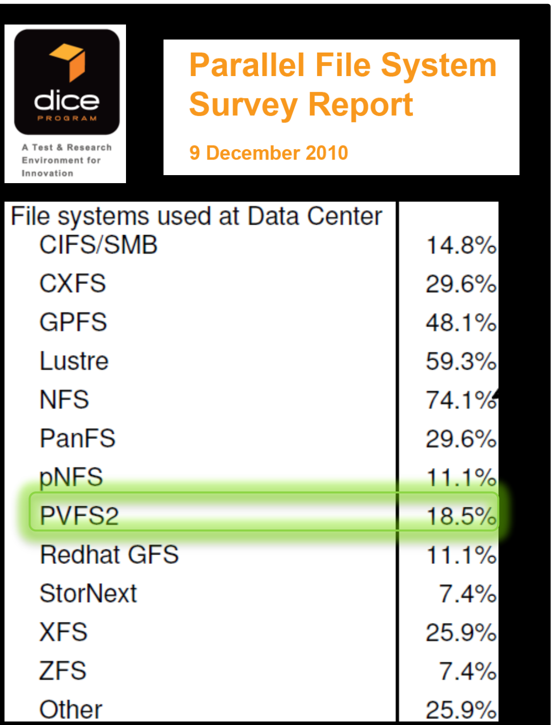
OrangeFS 2.8.5, 2.8.6, 2.9.0

Toward Exascale

OrangeFS NEXT

Future

PxFS



dice PROGRAM
A Test & Research Environment for Innovation

Parallel File System Survey Report

9 December 2010

File systems used at Data Center	
CIFS/SMB	14.8%
CXFS	29.6%
GPFS	48.1%
Lustre	59.3%
NFS	74.1%
PanFS	29.6%
pNFS	11.1%
PVFS2	18.5%
Redhat GFS	11.1%
StorNext	7.4%
XFS	25.9%
ZFS	7.4%
Other	25.9%

Original PVFS Design Goals

- Scalable
  - Configurable file striping
  - Non-contiguous I/O patterns
  - Eliminates bottlenecks in I/O path
  - Does not need locks for metadata ops
  - Does not need locks for non-conflicting applications
- Usability
  - Very easy to install, small VFS kernel driver
  - Modular design for disk, network, etc
  - Easy to extend

OrangeFS Philosophy

- Focus on a Broader Set of Applications
- Customer & Community Focused
- Embrace Research
- Completely Open Source
- Commercially Viable

FEATURES

System Architecture

- OrangeFS servers manage objects
  - Objects map to a specific server
  - Objects store data or metadata
  - Request protocol specifies operations on one or more objects
- OrangeFS object implementation
  - Berkeley DB for indexing key/value data
  - Local file system for stream of bytes

Semantics

- As we approach Exascale the reality of the limits of Sequential Consistency become more questionable. They are:
  - Expensive to implement for performance and scalability
  - Not needed if applications are well behaved
- OrangeFS uses a scalable lockless consistency model
  - Indistinguishable from SC for many programs
  - Provides much better performance/scalability
  - If the application requires locks, OrangeFS supports Distributed Lock Managers or application level locking, ex. Webdav interface for OrangeFS implements in metadata.

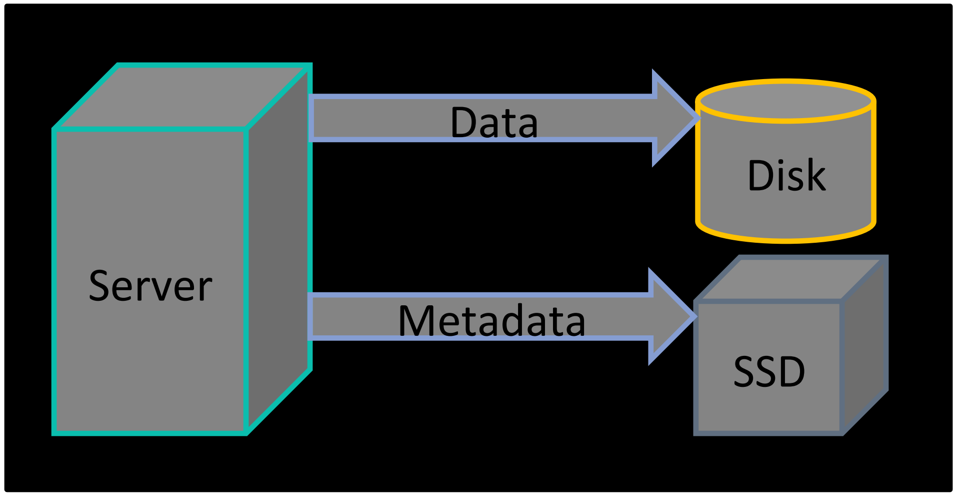
Recent Focus Areas

- Metadata Access Performance
- Reliability At Scale
- Security
- Diverse Access Methods
- Configurable Features
  - Avoid performance penalty for unused features

Recently Added to OrangeFS

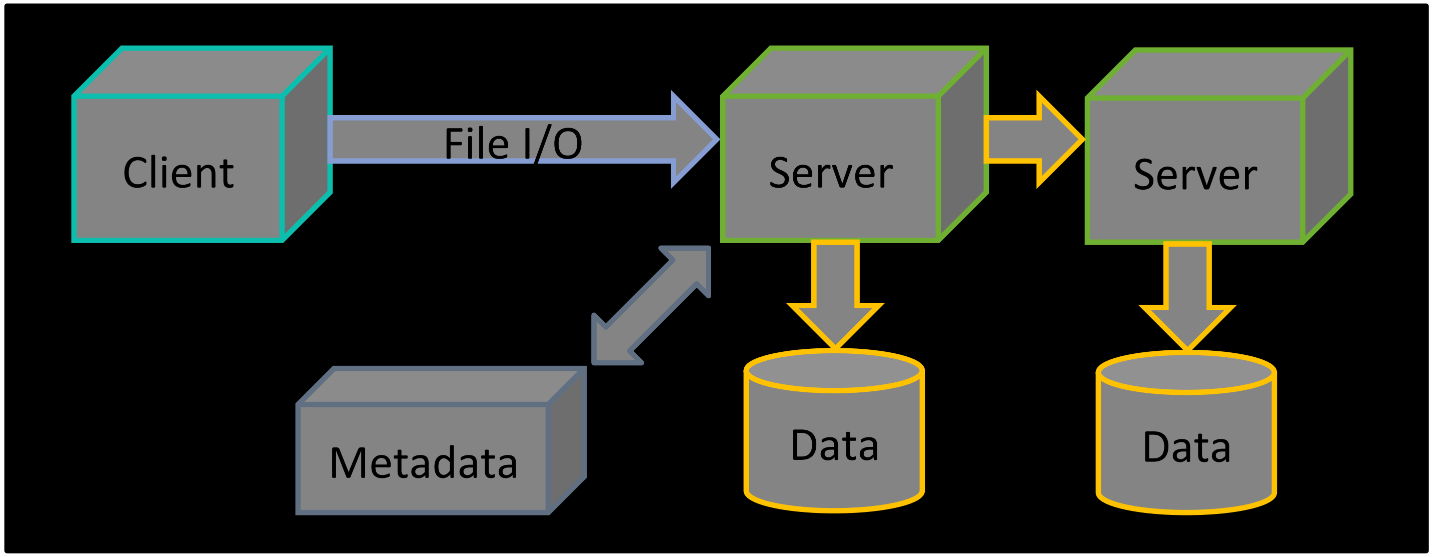
- In 2.8.3
  - Server-to-Server Communication
  - SSD Metadata Storage
  - Replicate on Immutable
- 2.8.4, 2.8.5 (fixes, support for newer kernels)
- Windows Client
- In 2.9.0 (1st half 2012)
  - Distributed Metadata for Directory Entries
  - Capability-Based Access Control
  - Direct Access Libraries
    - preload library for applications
    - Including Optional Client Cache

SSD Metadata Storage



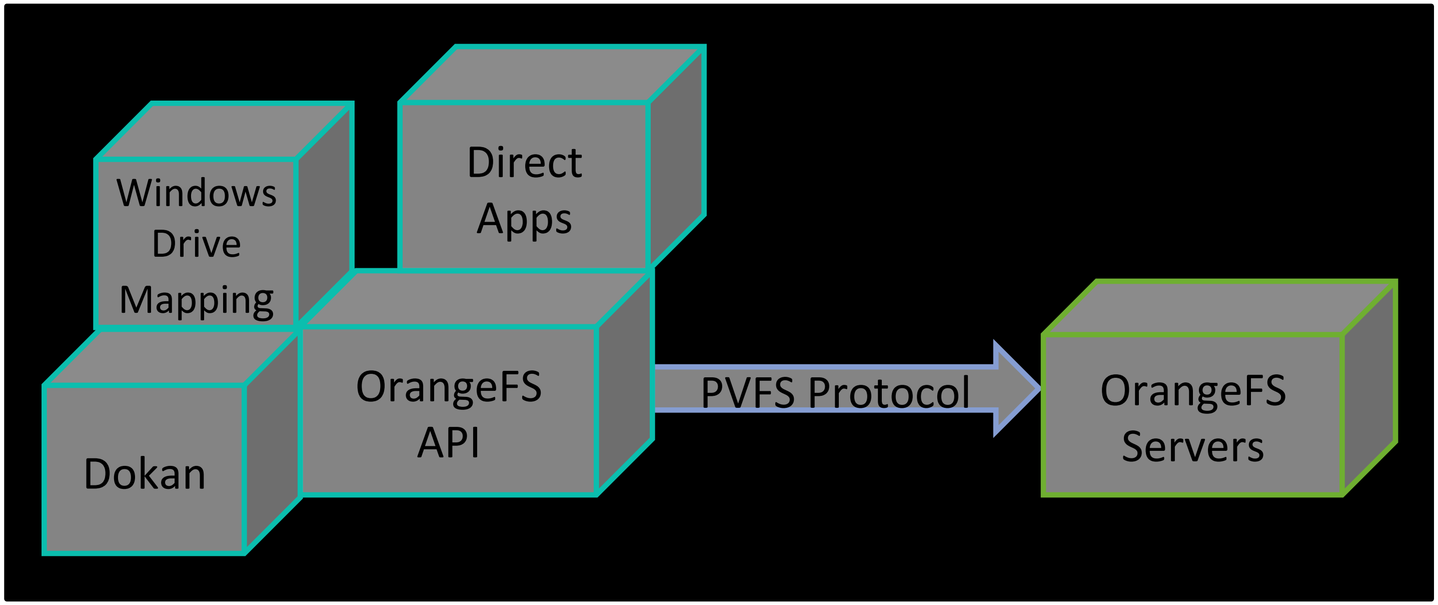
- Writing metadata to SSD
  - Improves Performance
  - Maintains Reliability

Replicate On Immutable



- First Step in Replication Roadmap
- Replicate data to provide resiliency
  - Initially replicate on Immutable
  - Client read fails over to replicated file if primary is unavailable

Windows Client

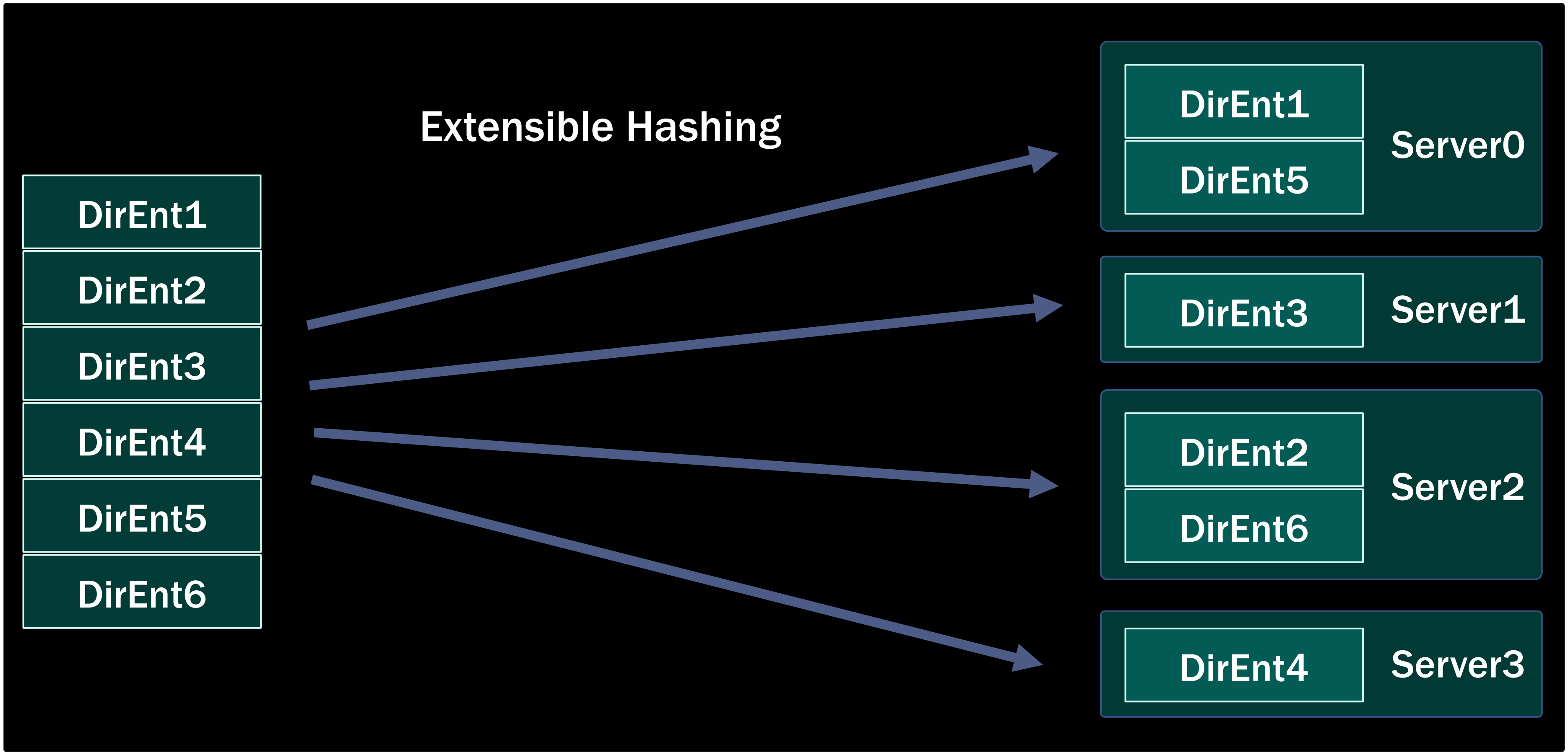


- Supports Windows 32/64 bit
- Server 2008, R2, Vista, 7

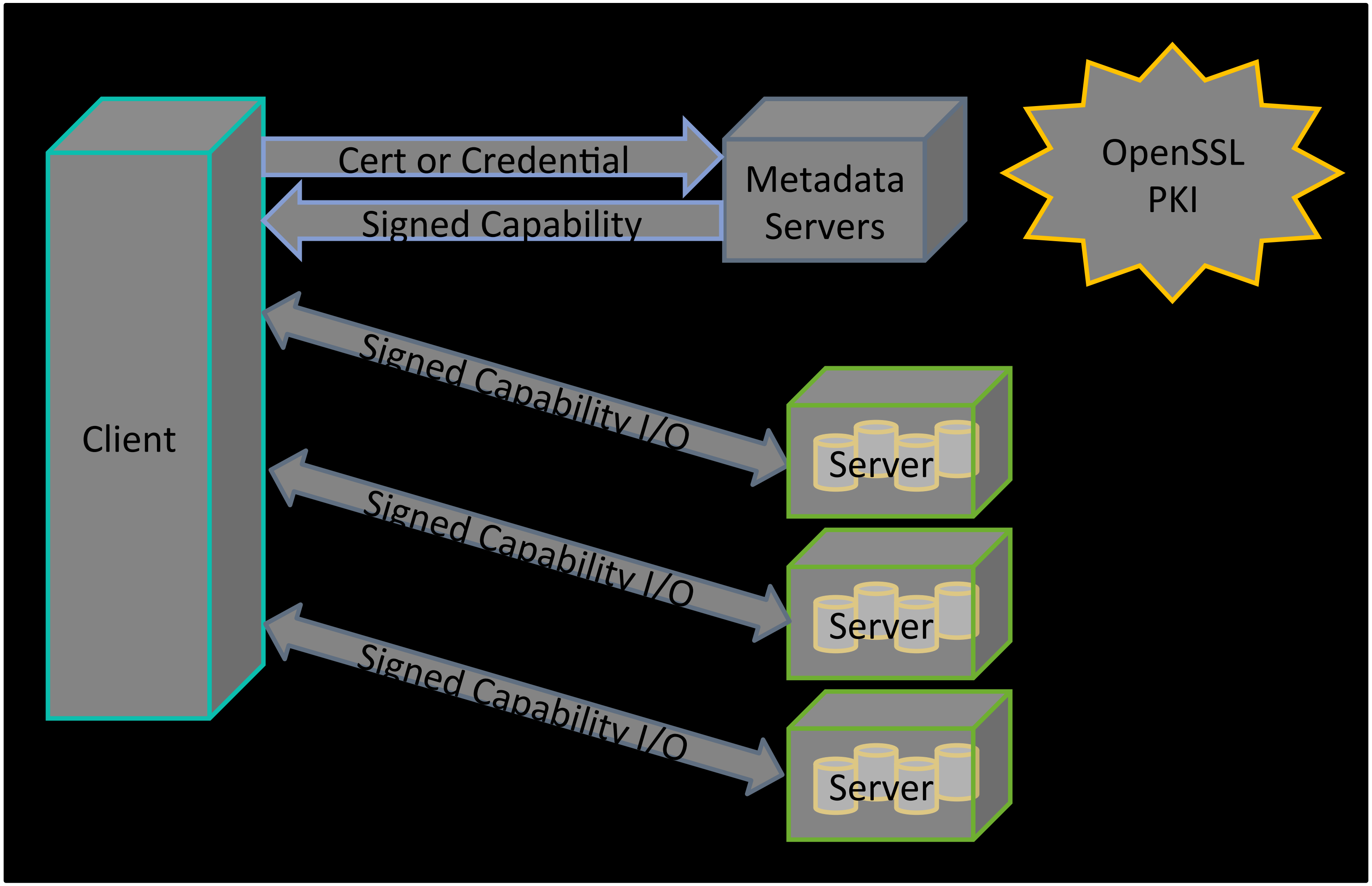
Coming in 2.9.0

- In 2.9.0 (1st half 2012)
  - Distributed Metadata for Directory Entries
  - Capability-Based Access Control
  - Direct Access Libraries (preload library for applications)

Distributed Directories



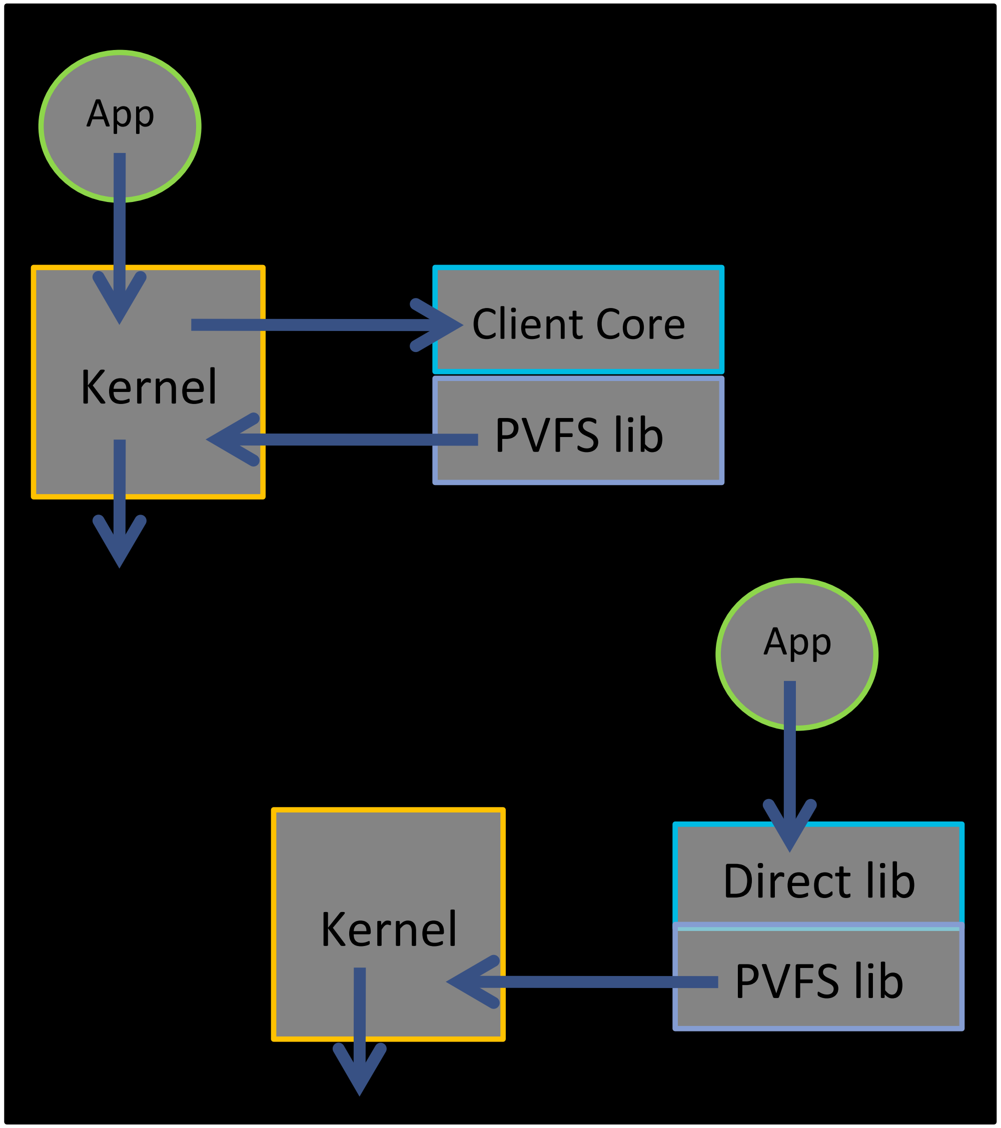
Capability Based Security



Interfaces

- Windows client (available now)
- FUSE (available now, improved mac support in 2.8.6)
  - Wider range of clients
  - Better stability using broader community code
- Web Package (Apache Modules) (2.8.6)
  - WebDAV Client
  - S3
  - REST Admin (DojoToolkit UI)
- Direct Access Libraries (2.9.0)
  - Better performance by bypassing kernel
  - Access OrangeFS and other files
  - Interface extensions
  - Easy to preload or link to applications

Direct Access Interface

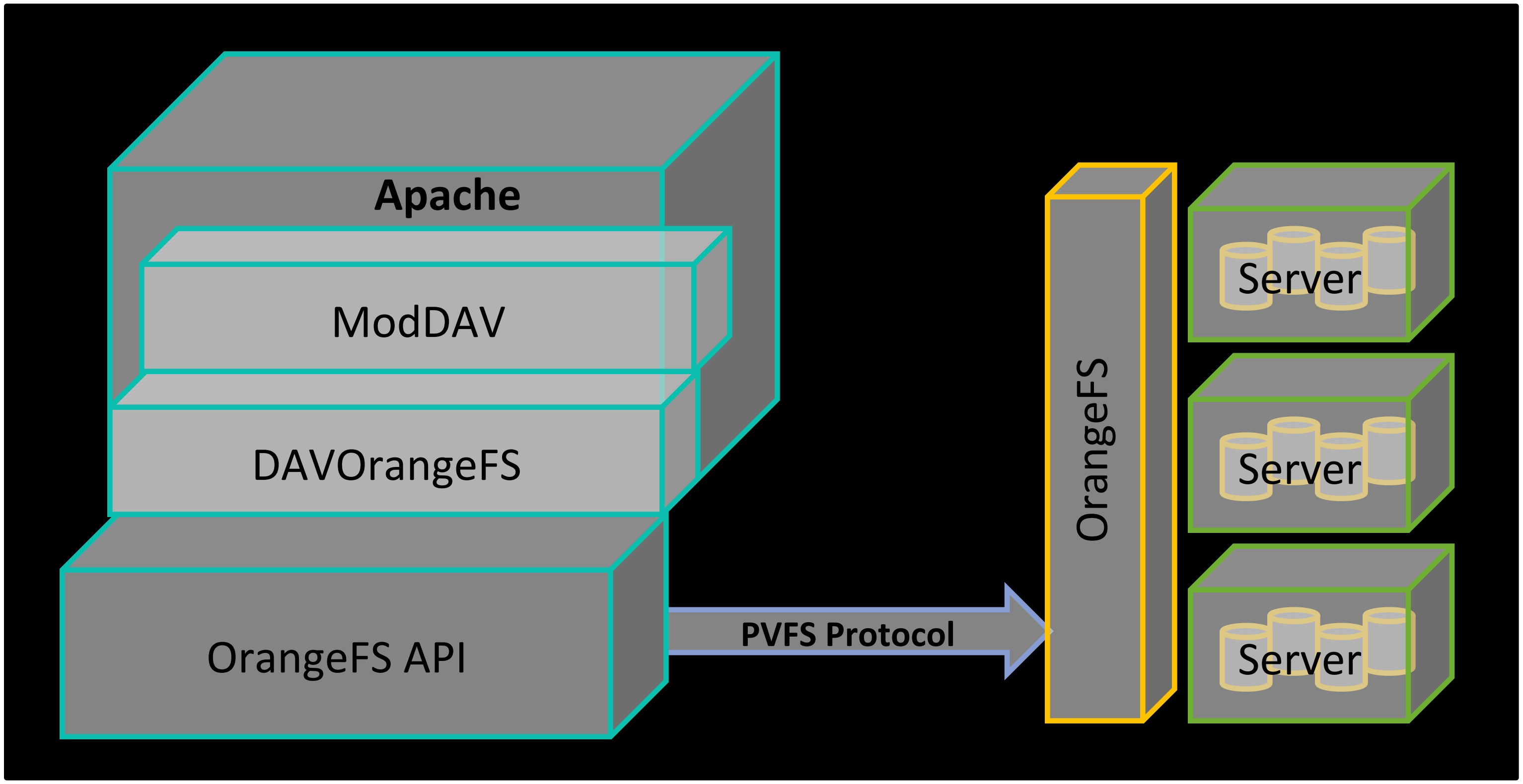


- Implements:
  - POSIX system calls
  - Stdio library calls
- Parallel extensions
  - Noncontiguous I/O
  - Non-blocking I/O
- MPI-IO library

Web Package

- WebDAV (2.8.x)
- S3 (2.8.x)
- REST Admin Interface (2.9.x)
- DojoToolkit Admin UI (2.9.x)

WebDAV



- Supports DAV protocol and tested with (insert reference test run – check with mike)
- Supports DAV cooperative locking in metadata

FUTURES

OrangeFS $_{\text{NEXT}}$

- 4 Foundational Elements of NEXT
  - File Handles –> 128bit UUID
  - Server Location and SID (Server Identifier) Management
  - Policy Based Configurable Replication, Migration and Hierarchical Storage
  - Attribute Based Metadata Search

File Handles -> UUID

- Currently use a 64 bit object handle space
  - Statically divided between servers
  - Defined in global configuration
- Going to 128 bit UUID object handles
  - Can be locally generated
  - Not known to all nodes
- Server ID identifies location of copies
  - May be stale

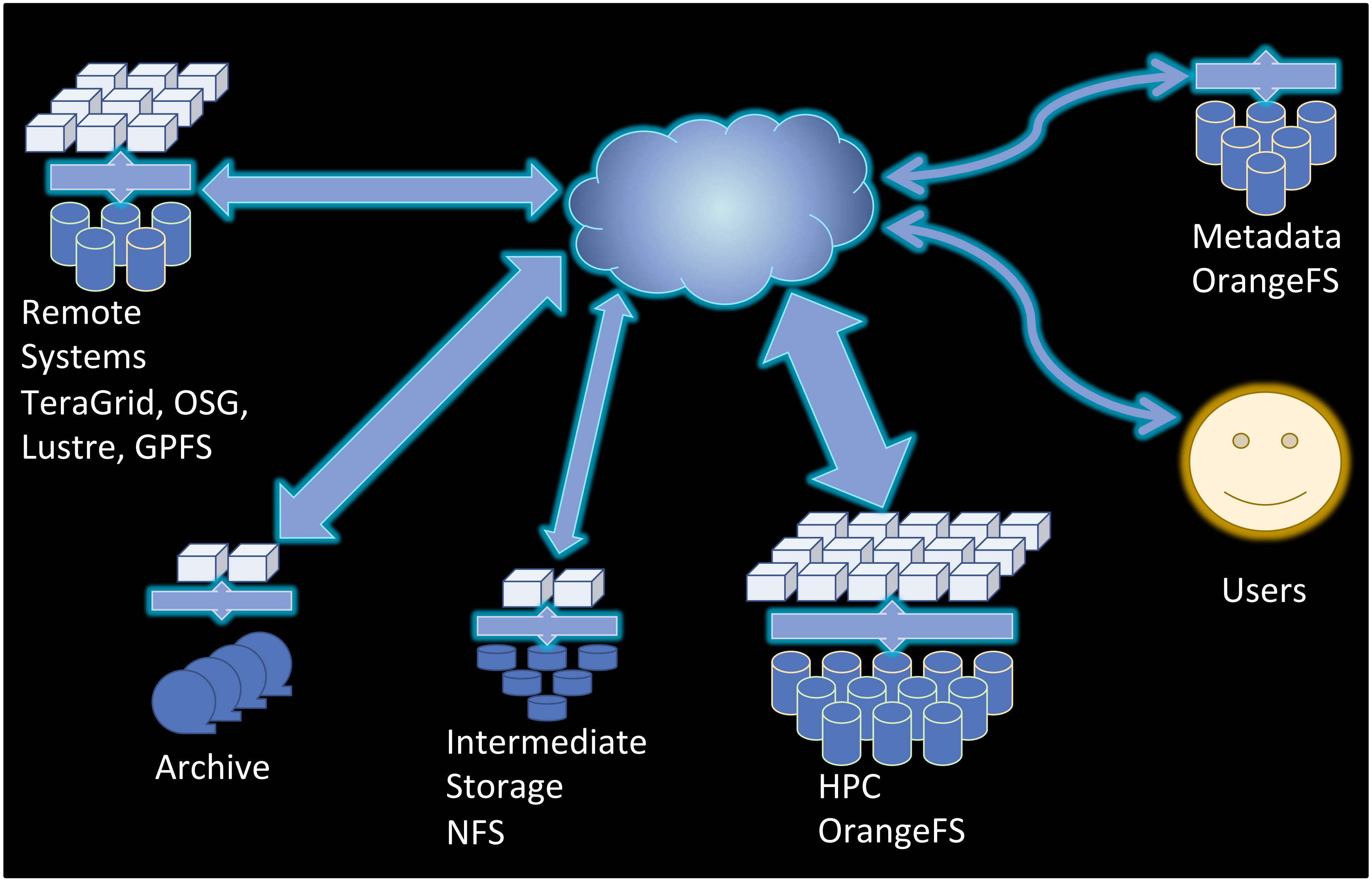
Replication / Redundancy

- Redundant Metadata
  - Easier recovery after a crash
  - Redundant objects from root directory down
  - Configurable
- Fully Redundant Data
  - Experiments with “forked flow” show small overhead
  - Configurable
    - Number of duplicates (O .. N)
    - Update mode (continuous, on close, on immutable, none)
- Emphasis on continuous operation

Migration

- Migrate objects between servers
  - De-populate a server going out of service
  - Populate a newly activated server
- Based on redundancy technology
  - Make a copy, then remove the old one
- Hierarchical storage
  - Use existing metadata services

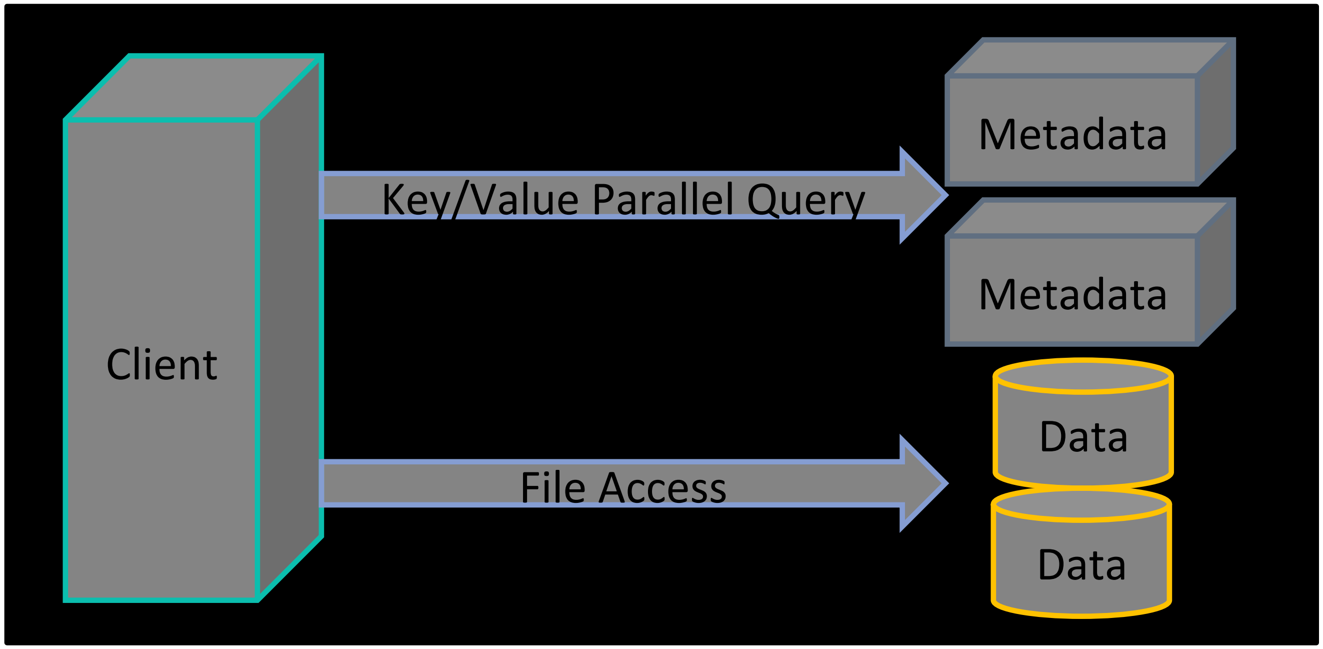
Hierarchical Data Management



Building on Replication

- OrangeFS metadata is extremely flexible
- Moving data
  - Migration (including hardware lifecycle)
  - Archival
  - Data staging
- Locating data
  - Within a directory
  - Across directories
  - Across devices
  - Across systems
  - Across Regions
- Moving computation to data

Attribute Based Metadata Search



- Client tags files with Keys/Values
- Keys/Values indexed on Metadata Servers
- Clients query for files based on Keys/Values
- Returns file handles with options for filename and path

Beyond OrangeFS $_{\text{NEXT}}$

- Extend Capability based security
  - Enables certificate level access
  - Federated access capable
  - Can be integrated with rules based access control
    - Department x in company y can share with Department q in company z
      - rules and roles establish the relationship
      - Each company manages their own control of who is in the company and in department

ParalleX

- Parallel Execution model
  - Shared Address Space (Shared Memory)
  - Communicating Sequential Processes (Message Passing)
  - ParalleX
- Reference implmentation
  - HPX (based on C++)

ParalleX Model

- Design Philosophy
  - Move work to data (message driven)
  - Local rather than global synchronization
  - Latency hiding with threads
- Key Components
  - Threads (lightweight)
  - Parcels (active messages)
  - Asynchronous Global Address Space (AGAS)
  - Local Control Objects (futures)
  - Processes (span locales)

PXFS

- Research project developing a parallel file system for ParalleX, based on OrangeFS.
- Key concept is unifying the namespace of the file system with the ParalleX AGAS
- Persistent objects can be moved in and out of storage as needed
- Futures allow massively parallel FS operations without costly global synchronization

PXFS Development

- Shares interesting problems with OrangeFS Next development
  - – Highly distributed name space (metadata)
  - – Replication, migration
- Research project under way
  - – LSU
  - – Clemson U
  - – Indiana U

COMMUNITY

Learning More

- www.orangefs.org web site
  – Releases, Documentation, Wiki
- pvfs2-users@beowulf-underground.org
  – Support for users
- pvfs2-developers@beowulf-underground.org
  – Support for developers
- www.orangefs.com & www.omnibond.com
  – Professional Support & Development team

Questions & Answers